IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received July 29, 2019, accepted August 13, 2019, date of publication August 21, 2019, date of current version September 4, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2936582

Accountable Outsourcing Location-Based Services With Privacy Preservation

ZHAOMAN LIU[1,2], LEI WU[1,2,3], JUNMING KE[4], WENLEI QU[1,2], WEI WANG[1,2], AND HAO WANG[1,2]

[1]School of Information Science and Engineering, Shandong Normal University, Jinan 10445, China
[2]Shandong Provincial Key Laboratory for Novel Distributed Computer Software Technology, Jinan, China
[3]Shandong Provincial Key Laboratory for Software Engineering, Jinan, China
[4]Information System Technology and Design Pillar, Singapore University of Technology and Design, Singapore

Corresponding author: Lei Wu (wulei@sdnu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61602287 and Grant 61672330, in part by the Primary Research and Development Plan of Shandong Province under Grant 2018GGX101037, and in part by the Major Scientific and Technological Innovation Project of Shandong Province under Grant 2018CXGC0702.

ABSTRACT With the enhancement of the positioning function of mobile devices and the upgrade of communication networks, location-based service (LBS) has become an important application of mobile devices. Among the numerous researches on location privacy preservation, cloud-based location privacy preservation has become a hot topic, but it undoubtedly brings new problems such as data confidentiality and user privacy disclosure. This paper proposes an accountable outsourced LBS privacy-preserving scheme. In the outsourcing scenario, in order to make users interact with cloud server to obtain query data, firstly we construct location hierarchical index and attribute hierarchical index based on Bloom Filter, and secondly we divide one region into atomic regions using Hilbert Curve, both of which ensure the privacy of query and improve the efficiency of query. At last, we realize the sharing of encrypted data among different users by accountable proxy re-encryption (APRE) technology, which can effectively suppress the abuse of proxy re-encryption key. We demonstrate the correctness of the proposed scheme through security analysis, and show the effectiveness of the scheme through performance analysis.

INDEX TERMS LBS, privacy preservation, Hilbert curve, proxy re-encryption, accountability.

I. INTRODUCTION

With the development of wireless communication technology and mobile positioning technology, more and more mobile devices have the precise positioning function of GPS, which makes the location-based service (LBS) increasingly popular. Location service provider provides services for users according to the location of mobile devices and other information [1], and its typical applications include map applications (such as Google Maps), point-of-interest(POI) retrieval (such as AroundMe), perimeter recommendations (such as GroupOn), GPS navigation (such as TomTom) and location-aware social networks (such as Foursquare) etc [1]. Although LBS brings great benefits to individuals and society, it also raises serious privacy concerns. Users need to report their location information while obtaining LBS, and the location data not only directly contains users' privacy information, but also implies other personal sensitive information that users usually want to protect, such as home address, lifestyle, health status and social relations. Therefore, the disclosure of such private information to untrusted third parties (such as LBS providers, LBSP) will open the door to abuse of personal data and pose a serious threat to the privacy of users in all aspects.

In order to lighten its burden, LBSP usually requests help from cloud service provider (CSP). LBSP can outsource its own data and query processing to cloud server, which liberates LBSP from complicated data operations. Although CSP can lighten the burden of LBSP, it also poses a number of threats. On the one hand, the CSP is considered to be honest but curious. It will honestly execute user's query requests. At the same time, it also wants to obtain valuable information from user's queries and LBSP's data. Therefore, the user usually encrypts the query content, and LBSP also stores the data on cloud server in the form of ciphertext, which preserves the privacy of user's query and the confidentiality of LBSP's data to a certain extent. On the other hand, when the data is corrupted, CSP may send damaged data to the user

The associate editor coordinating the review of this article and approving it for publication was Longxiang Gao.

in order to lighten its burden. This is clearly not what users and LBSP want to see. The above situation can be avoided by introducing integrity verification mechanism.

In the application scenario of surrounding recommended, the user requests points of interests (POIs) with a certain attributes near somewhere, which can be either the location of the mobile user or the other location. For example, Alice queried the gynecology hospital within 1000m of her neighborhood. In this example, Alice's query location is her own position, the query attribute is "gynecology hospital", and the query range is "$Distance(loc_{Alice}, loc_{hospital}) \leqslant 1000m$". In this scenario, the user hopes to get accurate query results as well as protects his location and attributes privacy against disclosure to CSP and LBSP. It is well known that there is a contradiction between query accuracy and privacy. Therefore, improving the query accuracy under the premise of protecting the privacy of user has become a hot topic in the research of location-based privacy preservation.

In this paper, LBSP constructs a "country-province-city-region" location hierarchical index, and uses the Hilbert curve to further divide the region into atomic regions, and stores the data according to the atomic regions which they belong to. An attribute hierarchical index is established for each atomic region. In this way, we can find the data of an atomic region according to the location hierarchical index and the atomic region number, and further find out the POIs by the attribute index of the atomic region. Location hierarchical index and attribute hierarchical index are constructed by Bloom Filter with key-hash functions, the advantage of which is that it is difficult to infer the content of the data from the hash value.

LBSP entrusts the query processing and outsources the resulting data to CSP in encrypted form. In order to obtain the result data, we use the accountable proxy re-encryption (APRE) mechanism to realize ciphertext sharing among authorized users. LBSP acts as the delegator, CSP acts as the proxy, and the user acts as the delegatee. First, LBSP generates the re-encryption key and sends it to the CSP, and then CSP uses it to re-encrypt ciphertext for the authorized user. The accountability mechanism can prevent CSP from converting all LBSP's ciphertext into user's, thus alleviate the abuse of re-encryption key.

The main work and contribution of this paper are as follows:

(1) We construct a LBS privacy-preserving scheme for query outsourcing, and generate the location index and attribute index using Bloom Filter with key-hash function. The scheme not only protects the location privacy and query privacy of users, but also ensures the implementation of query processing between users, LBSP and CSP.

(2) We establish a "country-province-city-region" hierarchical index, and use the Hilbert curve to divide the region into atomic regions. The data is stored in atomic regions to support the range query of POIs.

(3) We combine the accountable proxy re-encryption (APRE) technology with data outsourcing to realize the sharing of encrypted data among authorized users. The accountability mechanism can prevent malicious proxy conspiring with unauthorized user to obtain unauthorized data, which ensures the confidentiality of data.

The rest of the paper is organized as follows. Section 2 introduces the relevant work on LBS privacy preservation. Section 3 introduces the background knowledge used in this paper. The system model and the data storage model are depicted in Section 4, and the query model and potential threat model are also described in detail here. Section 5 describes the scheme of this paper in detail. The security analysis and performance analysis are showed in Section 6 and Section 7 respectively. In Section 8, we evaluate the efficiency of the tools used in this paper and finally we make a conclusion in Section 9.

II. RELATED WORK

LBS privacy preservation has been proposed for many years. At present, the work of LBS privacy preservation is classified as follows: privacy preservation based on policy, privacy preservation based on obfuscation, privacy preservation based on disturbance, and privacy preservation based on encryption. Table 1 gives a detailed comparison of different classification methods.

A. POLICY-BASED LBS PRIVACY-PRESERVING TECHNOLOGY

Policy-based technologies rely primarily on economic, social and regulatory pressures to enable that LBS providers can use location information in queries fairly and securely in accordance with privacy management rules and trusted privacy protocol. The geographic location privacy (GeoPriv) of IETF [2] and platform for privacy preferences (P3P) of W3C [3] are typical representative technologies.

B. OBFUSCATION-BASED LBS PRIVACY-PRESERVING TECHNOLOGY

Obfuscation refers to the generalization or disturbance of the time or location in LBS query $< u, t, loc, U_{POI} >$, which makes it impossible for an attacker to identify the query user and get the exact location. Generalization refers to reducing the accuracy of query location or query time in a controllable way. It is usually replaced by a region (spatial generalization) or a period (temporal generalization), so that a certain number of users and query user can share the same $< t, loc >$. Disturbance refers to introduce some error locations intentionally in a controllable way, such as replacing the real location with anchors, simultaneously, the data still need to meet the requirements of service quality [4].Obfuscation-based technologies mainly include K anonymity and differential privacy.

1) K ANONYMITY

Spatial generalization technologies such as Interval Cloak [5], Casper Cloak [6], Clique Cloak [7], and Hilbert Cloak [8] are designed to protect query privacy, so that an attacker cannot infer which user performs a query among k different users.

In addition to location K anonymity, Huo *et al.* [9], Gao *et al.* [10], Wu *et al.* [11] propose trajectory K anonymity. Moreover, in the aspect of trajectory prediction, [12] designs a K anonymous protection algorithm based on probability inference model Mask-K. According to the background knowledge of the attacker, Lee *et al.* [13] propose a dynamic K anonymity algorithm to resist the attack based on historical trajectory prediction. Lin and Wang [14] calculate the trajectory similarity by EMD algorithm, which is used to measure the level of privacy leakage.

2) DIFFERENTIAL PRIVACY

Differential privacy is a new definition of privacy security proposed by Dwork in 2006 [15]. It is as yet the most effective technology to provide universal preservation against prior knowledge of attackers. At present, it is mainly used in location privacy preservation of data publishing. In the field of location data, the research of differential privacy has been related to specific multidimensional index quadtree [16], [11], [18]. [19] proposes an attack model aiming at background knowledge that can be applied to differential privacy. Zhang *et al.* [20] propose a location big data publishing algorithm (Diff_Anonymity) based on differential privacy, which can reduce the risk of privacy disclosure. Bi *et al.* [21] propose a location privacy-preserving method of quadratic anonymity based on privacy preference, which combines K anonymity and differential privacy. It can not only ensure that the points in anonymous set have the maximum probability similarity, but also realize personalized setting of privacy-preserving level. The concept of "region indistinguishability" is introduced by GPOL [22], and a new LBS group nearest neighbor query method based on differential privacy is proposed. The nearest neighbor query of the group is transformed into the query of the centroid, which can effectively resist the existing cross-attack and collusion attack.

C. DISTURBANCE-BASED PRIVACY-PRESERVING TECHNOLOGY

Disturbance means that using a pseudonym, a false location, or a false query instead of user's real u, loc, or U_{POI} in a query $< u, t, loc, U_{\mathrm{POI}} >$.

1) PSEUDONYM

Mix-zone is a way to achieve anonymity [23], [24]. Although the hybrid area weakens the attacker's ability to associate the user's new pseudonym with the old one, if the attacker considers the user's mobility model, the effectiveness is significantly reduced. Blurring the time or place when or where the pseudonym changes can enhance the effectiveness of the hybrid area. For example, Palanisamy et al. [25], [26] propose to establish an irregular Mix-zone model at the intersection of a real road network. In order to resist the attacker's timing attack, a random transformation of the real request time is carried out. In [27], by adding the time window to improve the security of Mix-zone, the problem of privacy disclosure in continuous query is solved. Liu *et al.* [28], Liu and Li [29] design a multi-granularity hybrid area deployment scheme. Taking into account both population density and traffic variability, a new metric is developed to measure the privacy-preserving capability of the system. Sun *et al.* [30] apply Mix-zone technology to the deployment of urban road network, and propose a statistical-based scheme to minimize the Mix-zone.

2) FALSE LOCATION AND FALSE QUERY

[31] considers indicators such as universality, congestion, and uniformity to produce dummy positions similar to those of real users' mobility. Whereas the dummy positions' movement features in historical data are very different from those of real user, and some dummy positions may even be at positions that are not available in practice (for example, in the ocean), which is easy to be identified by an attacker. SpaceTwist [4] randomly selects the anchor point instead of the user's real location to initiate multiple rounds of incremental nearest neighbor queries to the server. Although SpaceTwist avoids the high computation and communication overhead caused by the region query, the degree of privacy preservation is not high. Randomized technology is also used to protect query privacy, for example, the dummy position generated by DUMMY-Q method [32] is $\mathrm{U_{POI}}$, which takes the query context and the user's mobile model into account.

D. ENCRYPTION-BASED PRIVACY-PRESERVING TECHNOLOGY

Encryption-based technology achieves the purpose of privacy preservation by encrypting queries to make them completely invisible to the server. It is mainly divided into the technology based on privacy information retrieval (PIR) [33]–[36] and the method based on Hilbert curve encryption [37], [38]. At present, the LBS privacy-preserving technology based on PIR mainly aims at the privacy disclosure problem in the nearest neighbor (NN) query [33], [34], [39], [40] and shortest path calculation [35]. In [35], PIR protocol is used to study the privacy problem of shortest path query in traffic network. By proposing a concise index strategy, reasonable retrieval time is obtained and strong privacy preservation of shortest path calculation is realized. However, the time-space overhead caused by PIR is still much higher than that of unprotected query processing. It is a possible solution to compress network data by considering the characteristics and structure of data [41].

The position anonymity method HilCloak based on Hilbert Curve is proposed by [37], which rotates the whole space at an angle and sets up the Hilbert curve with the key H in the rotated space. Reference [42] combines hyperbolic query analysis technology and encrypted hash function to enhance the location privacy of HilCloak against attackers with strong prior knowledge and reduce query complexity.

In [38], a new spatial transformation method is proposed. The transformation of location data adopts the adaptive Hilbert curve, which changes with the distribution of POIs and achieves multi-granularity privacy preservation. Reference [43] proposes spatial query on outsourced private data and password conversion scheme based on Hilbert curve, which protects the privacy of spatial data against all kinds of attacks and is robust to the attack model. Liang *et al.* [44] combine K anonymity with Hilbert curve and propose a privacy-preserving method for the query of POIs in the road network environment, which can effectively resist the inference attacks.

III. PRELIMINARIES

This section provides a brief overview of the fundamental techniques used in the proposed scheme.

A. BLOOM FILTER

Bloom Filter (BF) is a random data structure with high spatial efficiency. It makes use of bit array to express a set succinctly, and can judge whether an element belongs to the set or not. Bloom Filter may make a mistake, but it will not miss the judgment. In other words, if Bloom Filter determines that an element is not in the set, then that element is definitely not in the set. If it decides that the element is in the set, then there's a certain probability that it's wrong. Bloom Filter consists of two important components. One is a bit array of length m, the other is the k independent hash functions $hash_i(x)$ $(1 \leqslant i \leqslant k)$, which maps each element in the set to $\{1, \dots, \mathrm{m}\}$. When we add any element x to Bloom Filter, we use k hash functions to get k hash values, and then set the corresponding bits in the array to 1. When determining whether element y belongs to this set, If all $hash_i(y)$ are 1 $(1 \leqslant i \leqslant k)$, then we consider y to be a member of the set, otherwise we consider y not to be a member of the set.

The error rate p of Bloom Filter is related to the length m of the bit array, the number of data items n, and the number of hash functions k. The error rate p can be expressed as $p = 1 - (e^{\frac{-(kn)}{m}})^k$. It achieves minimum value when $k = \ln 2 \cdot \frac{m}{n}$, which is $p = 2^{-\ln 2 \cdot \frac{m}{n}}$. Thus, when constructing Bloom Filter, given the number of elements n and the desired error rate p, the remaining parameters m and k can be automatically calculated.

B. HILBERT CURVE

The space filling curve can map the data in the high-dimensional space to the one-dimensional space. In this way, the adjacent objects in the space will be stored in one block adjacent to each other, which can reduce the time of input and output, as well as improve the efficiency of data processing in the memory. Hilbert curve can linearly penetrate each discrete unit in two-dimensional or higher dimensions once and only once according to its initial parameters and each discrete unit is linearly sorted and encoded. This code serves as the unique identity of the unit. The Hilbert curve defines the order of points on the two-dimensional plane. At the root level, once a direction and a starting point are

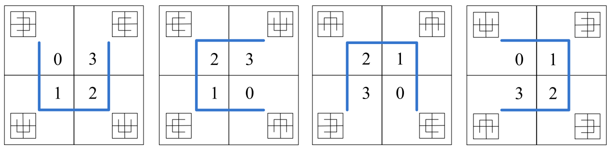


FIGURE 1. Four types of Hilbert Curve.

selected, the order of each quadrant can be determined by numbering them from 0 to 3. When we want to determine the order of subquadrant while maintaining the overall adjacency properties, we need to perform a simple transformation of the original curve. For a given quadrant, the curve in it is determined by the curve of the big square that the quadrant is in and the position of that quadrant. According to the direction of the previous layer, there are four transformations of the current layer as shown in Figure 1.

C. PROXY RE-ENCRYPTION

In 1998, Blaze M put forward proxy re-encryption for the first time [45]. Proxy re-encryption is a key conversion mechanism between ciphertexts. In proxy re-encryption, a semi-trusted proxy converts ciphertext encrypted with the public key $\mathrm{pk_A}$ of Alice (Delegator) into ciphertext decrypted with the private key $\mathrm{sk_B}$ of Bob (Delegatee) through the conversion key $\mathrm{rk_{A\to B}}$ generated by delegator. In this process, the proxy cannot get the clear information of the data, thus reducing the risk of data leakage. In this paper, we adopt the accountable proxy re-encryption technology (APRE) proposed by [46], which can not only realize the sharing of encrypted data among authorized users, but also publicly judge the distributor of malicious decryption right by the judge algorithm. The technique has the following key steps:

Setup(λ): Given a security parameter λ, output system parameter *param*.

ProxySet(*param*): Given a system parameter *param*, output proxy's key pair ($\mathrm{sk_{pro}}$, $\mathrm{pk_{pro}}$).

KeyGen(*param*): Given a system parameter *param*, output a key pair (sk_i, pk_i).

ReKeyGen($\mathrm{sk_p}$, pk_i, $\mathrm{pk_{pro}}$): Given delegator's secret key $\mathrm{sk_p}$, delegatee's public key pk_i, and proxy's public key $\mathrm{pk_{pro}}$, output re-encryption key $\mathrm{rk}_{\mathrm{p}\to i}$.

Enc$_1$(pk_i, m): Given delegatee's public key pk_i and message m, output the first level ciphertext C_i'.

Enc$_2$($\mathrm{pk_p}$, m): Given delegator's public key $\mathrm{pk_p}$ and message m, output the second level ciphertext C_p.

ReEnc($\mathrm{rk}_{\mathrm{p}\to i}$, $\mathrm{sk_{pro}}$, C_p): Given re-encryption key $\mathrm{rk}_{\mathrm{p}\to i}$, proxy's secret key $\mathrm{sk_{pro}}$ and the second level ciphertext C_p, output re-encrypted ciphertext C_i'.

Dec$_1$(sk_i,C_i'): Given delegatee's secret key sk_i and the first level ciphertext C_i', output the message m.

Dec$_2$($\mathrm{sk_p}$, C_p): Given delegator's secret key $\mathrm{sk_p}$ and the second level ciphertext C_p, output the message m.

Judge$^{D_{\mathrm{p},u}}$($\mathrm{pk_p}$, $\mathrm{pk_{pro}}$): Given the decryption device $D_{\mathrm{p},u}$, the delegator's public key $\mathrm{pk_p}$, and the proxy's public key $\mathrm{pk_{pro}}$, output proxy or delegator.

TABLE 1. Comparision of different LBS privacy-preserving techniques.

Category	Technique	Method	Degree of privacy preservation	Quality of Service
Policy-based		GeoPriv[2]	low	optimal
		P3P[3]	low	optimal
Obfuscation-based	K anonymity	Interval Cloak[5]	high	bad
		Casper Cloak[6]	high	average
		Clique Cloak[7]	high	well
		Hilbert Cloak[8]	high	well
Disturbance-based	Trajectory K anonymity	YCWA[9]	high	average
		Gao[10]	average	well
		Wu[11]	high	well
	Trajectory prediction	Mask-K[12]	high	well
		Li[13]	high	well
		Lin[14]	high	well
	Differential Privacy	Diff-Anonymity[20]	high	well
		GPoL[22]	high	well
	Pseudonym	Mix-zone[23-30]	average	well
	False position	Space Twist[4]	low	optimal
	False query	Dummy-Q[32]	average	optimal
Encryption-based	PIR[33-36]		high	optimal
	Hilbert curve Encryption	Hilcloak[37]	high	optimal

IV. MODEL

A. SYSTEM MODEL

As shown in Figure 2, the system consists of three entities: user, LBSP, CSP. In the pre-processing phase, LBSP constructs a hierarchical index of ''country-province-city-region'' and further divides the region into atomic regions. The data is stored in atomic regions and attribute hierarchical indexes are generated for each atomic region. In the initialization phase, the user registers with the LBSP, and the LBSP generates the identity id_i and key pairs (pk_i, sk_i) for re-encrypting and decrypting the data, and sends the key k_1, k_2 related to the key-hash function family in the Bloom Filter to the user. At the same time, LBSP and user share location index. When the user wants to query for POIs within a specific range of a location, the user first requests LBSP for the division of the region, and the LBSP sends the atomic regions of the region to the user. After that the user determines the query scope of the atomic regions, generates a query and sends it to the CSP. The CSP requests the LBSP for the region data that the user queries. Then, LBSP will send re-encryption key, the region data and attribute hierarchical indexes to the CSP. Once CSP matches query and index well, the result data will be re-encrypted by re-encryption key and returns to the user. Finally, the user decrypts the data with the private key and gets the query result.

B. STORAGE MODEL

LBSP sets up location hierarchical index and attribute hierarchical index, and conducts region division at the leaf node level of the location index. The depth of the index will have an impact on the query performance. We establish a location hierarchical index of ''country-province-city-region'', and further divide region into atomic region. The data is stored according to the atomic region, and the attribute index is established according to the attributes of the data in atomic region. Partition granularity affects the amount of data contained in each partition, which in turn affects the time and performance of query processing.

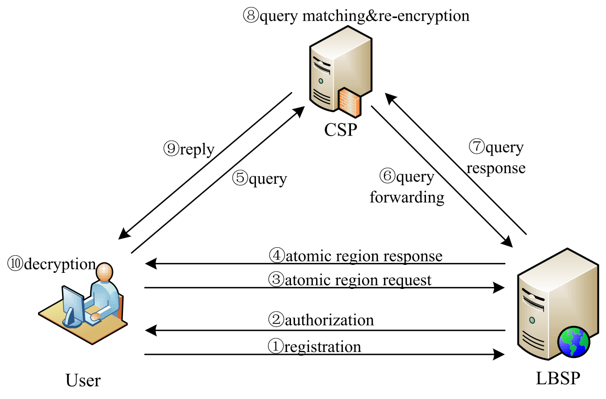


FIGURE 2. System model for outsourced LBS.

1) LOCATION INDEX CONSTRUCTION

In this paper, the hierarchical index of ''country-province-city-region'' is established by the method of bottom-up construction, where the root node represents the country and the leaf node represents the region. Taking the name of the region as the input of k hash functions associated with key k_1 in Bloom Filter, we will get k different bit positions. The value of the parent node is the union of its children'. Until the Bloom Filter value of the root node is generated, we finish the construction of location hierarchical index. Figure 3 depicts the construction of a location hierarchical index.

2) REGION DIVISION

After LBSP establishes the location hierarchical index, it will further use Hilbert curve to divide the region into atomic regions. The LBSP sets the threshold value s of the region division, and selects appropriate curve order for the different regions so that the width of the partitioned atomic region does not exceed the threshold s. For example, If LBSP sets the threshold s as $500m$, that is, the width of the atomic region generated by the Hilbert curve should not exceed $500m$. After the division of region, the region is represented as the set of

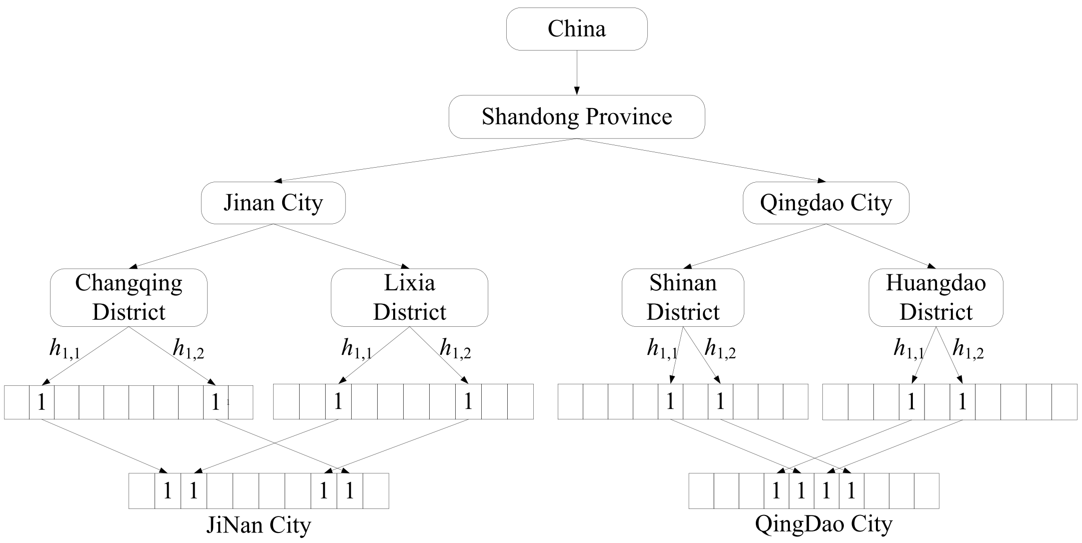


FIGURE 3. Location index construction.

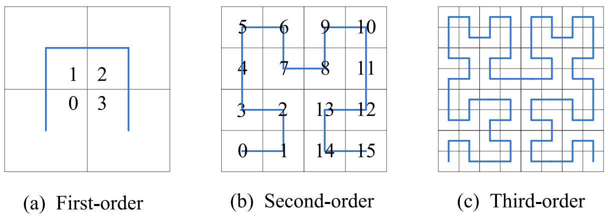


FIGURE 4. Region partition by Hilbert Curve.

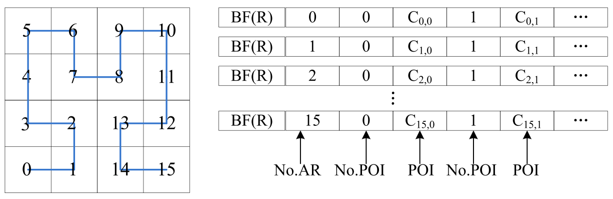


FIGURE 5. Storage structure of POI.

atomic regions, and the atomic regions are numbered with the Hilbert value of each atomic region. The specific division form is shown in Figure 4.

3) STORAGE STRUCTURE

LBSP stores points of interest distributed in the same atomic area in a linear structure connected at logical locations. The structure is composed of four parts: (1) region identity, (2) atomic region number, (3) POI number and (4) POI information, where the region identity is expressed by Bloom Filter value of the region name, and the information of POI are encrypted, which is shown in Figure 5.

4) ATTRIBUTE INDEX CONSTRUCTION

After dividing the region into atomic regions, LBSP constructs attribute index for data in each atomic region. Similar to location index, attribute index is also constructed using a bottom-up approach. The attribute hierarchical index maps attribute values into Bloom Filter using k hash functions associated with key k_2. The specific construction form of the attribute index is shown in Figure 6. Each node of the attribute index stores two parts: (1) the Bloom filter value of the attribute, and (2) the numbers of the POI corresponding to the attribute value, which is helpful to find the result data quickly. Figure 7 illustrates the storage of data and the construction of its attribute index in an atomic region.

C. QUERY MODEL

The query q generated by user contains the following parts: (1) the region R, (2) the set of atomic regions determined by query location and range $\{\mathrm{AR}_1, \mathrm{AR}_2, \ldots, \mathrm{AR}_n\}$, and (3) the attribute of POI *attr*. Before querying, the user firstly sends the region R with the identity id_i, public key pk_i to the LBSP, which aims to request the atomic regions of R. After receiving the information sent by the user, LBSP judges the legitimacy of the user according to id_i and pk_i. If the user's identity is legal, LBSP performs a query on the region R and sends its atomic region numbers to the user. The user selects the atomic regions according to the query range determined by himself, and sends the generated query q, identity id_i and pk_i to CSP. After receiving the query request sent by the user, the CSP forwards the region R in q, id_i, and pk_i to LBSP so as to request the data of the region R. In the location index, the root node represents the country; the leaf node represents the region. LBSP starts from the root node, and traverses through the leaf node layer by layer. Then, it will extract all the atomic region data and attribute hierarchical index of the leaf node from the database. Finally, LBSP sends atomic region dataset, attribute hierarchical index set, and re-encryption key $\mathrm{rk}_{\mathrm{p}\to i}$ to CSP. Once receiving the data of the region sent by LBSP, CSP performs attribute query on

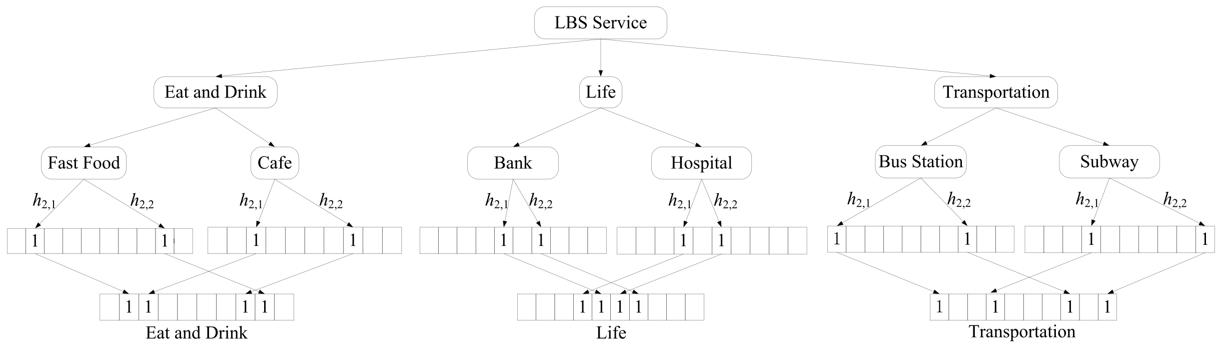


FIGURE 6. Attribute index construction.

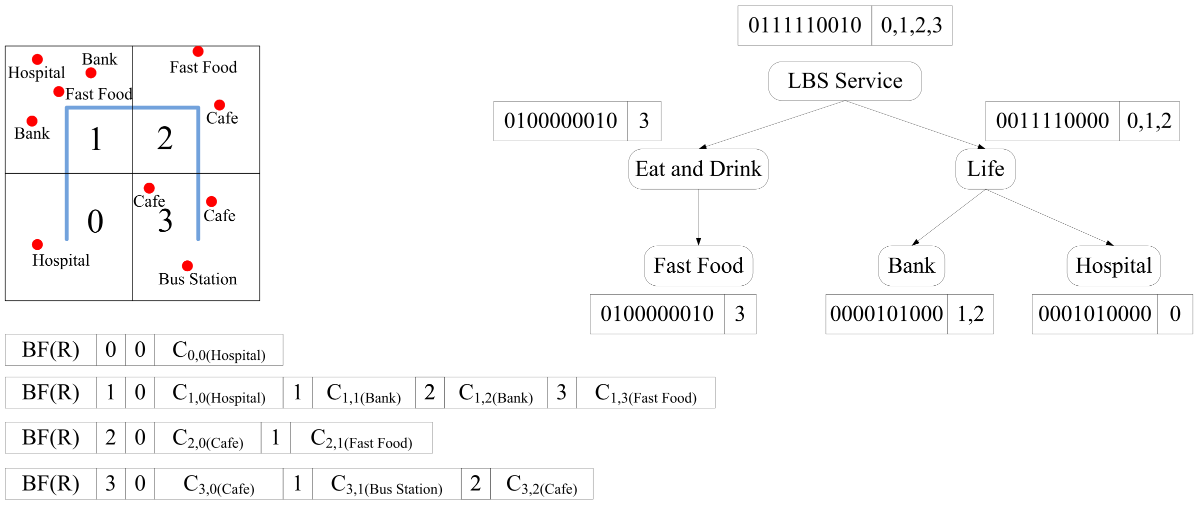

Hospital
Bank
Fast Food
Bank
Fast Food
Cafe
1
2
0
3
Cafe
Cafe
Hospital
Bus Station
0111110010 | 0,1,2,3
LBS Service
0100000010 | 3
0011110000 | 0,1,2
Eat and Drink
Life
Fast Food
Bank
Hospital
0100000010 | 3
0000101000 | 1,2
0001010000 | 0
BF(R) | 0 | 0 | $C_{0,0(\text{Hospital})}$
BF(R) | 1 | 0 | $C_{1,0(\text{Hospital})}$ | 1 | $C_{1,1(\text{Bank})}$ | 2 | $C_{1,2(\text{Bank})}$ | 3 | $C_{1,3(\text{Fast Food})}$
BF(R) | 2 | 0 | $C_{2,0(\text{Cafe})}$ | 1 | $C_{2,1(\text{Fast Food})}$
BF(R) | 3 | 0 | $C_{3,0(\text{Cafe})}$ | 1 | $C_{3,1(\text{Bus Station})}$ | 2 | $C_{3,2(\text{Cafe})}$

FIGURE 7. Attribute Index Construction of atomic region 1.

each atomic region of query q, starting from the root node of the attribute index to the last exactly matching node, and find out specific POIs according to POI numbers stored in the attribute node. After the CSP finds POIs that satisfy the query, it will use the re-encryption key $\mathrm{rk}_{p\rightarrow i}$ to re-encrypt the POI dataset, and the encrypted results will be sent to the user, then user uses his own private key to decrypt the final POI information.

D. THREAT MODEL

Our threat models are consistent with most other work in this area. CSP and LBSP are considered to be honest but curious. That is, they honestly follow the designed protocol, but at the same time they may analyze and infer the available data. LBSP can try to analyze the specific location of the user through the user's query region R. CSP can not only try to analyze the query q submitted by the user, and it can also analyze the encrypted data and attribute index held by LBSP, as well as the information of the data according to the POI numbers of each node in the attribute index. In addition, it is assumed that CSP and LBSP do not collude when attempting to collect user's information.

E. TRACKING THREATS

CSP can record queries because user may constantly send location queries. If CSP can learn any important information from the query, it may try to infer the approximate location of the user.

1) LINKAGE THREAT

Through the atomic region numbers submitted by the user, CSP may determine the location of the user by analyzing the construction of the Hilbert curve. CSP stores encrypted data and indexes and performs search operations. Based on which, CSP may attempt to link the query to the retrieved data.

V. SCHEME

In this paper, there are three entities involved in the system. They are user, LBSP, CSP respectively. The system is divided into several stages including setup, initialization, query generation, query processing, proxy re-encryption and decryption. Each stage will be described separately as follows.

A. SETUP

LBSP generates its own key pair $(\mathrm{pk}_p, \mathrm{sk}_p)$ for encrypting data. At the same time, LBSP randomly selects two number

Algorithm 1 $DBEnc(\mathrm{pk_p}, DB) \rightarrow C_{DB}$

```
1: for all $record \in DB$ do
2:     $C_{record} \leftarrow RecordEnc(record, \mathrm{pk_p})$
3:     $C_{DB} \leftarrow C_{DB} \cup C_{record}$
4: end for
5: return $C_{DB}$
```

k_1, k_2 for key-hash function groups $h_1()$ and $h_2()$, respectively. Each function in $h_1()$ maps a location to a bit in Bloom Filter, and each function in $h_2()$ maps an attribute to a bit in Bloom Filter. $h_1()$ and $h_2()$ are publicly available and assume that they include k hash function respectively. The input field of all hash functions is a binary string of length λ, where λ is a security parameter, and the output field is from 1 to the length of Bloom Filter.

After generating the above parameters, LBSP constructs the hierarchical pattern of location, and further divides every region R into atomic regions by using Hilbert curve, and stores the information of POIs distributed in the same atomic region together. Simultaneously, an attribute hierarchical index is constructed for each atomic region. Then, LBSP uses its own public key $\mathrm{pk_p}$ to encrypt the database DB. Algorithm 1 describes the encryption process, where *RecordEnc* is an algorithm for encrypting every POI data.

In addition, CSP sends its own public key $\mathrm{pk_{pro}}$ to LBSP as a parameter for generating a proxy re-encryption key. In this paper, CSP is regarded as a proxy in the process of re-encryption.

B. INITIALIZATION

The user registers with LBSP, and LBSP generates identity id_i, key pair $(\mathrm{pk}_i, \mathrm{sk}_i)$ for the user, and sends two keys k_1, k_2 related to the key-hash functions to the user. The parameter sent to the user by LBSP during the initialization phase is $param = \{id_i, \mathrm{pk}_i, \mathrm{sk}_i, k_1, k_2\}$.

C. QUERY GENERATION

After receiving the k_1, k_2 sent by LBSP, the user takes k_1 and the region R that contains query location loc as the input of the key-hash function group $h_1()$ and obtains k hash values, then sets the bit position corresponding to the hash values as 1. Eventually, the user gets the Bloom Filter value qv_R of the location loc. Similarly, the query attribute $attr$ and k_2 are entered into the key-hash function group $h_2()$, and the Bloom Filter value qv_{attr} of the attribute $attr$ is obtained. Before the query is generated, the user sends (id_i, pk_i, R) to LBSP to request atomic region encoding of the region R. If region R exists, the LBSP sends the encoding to the user. Then the user sets the query range ran, and takes the atomic regions covered by ran as part of the query q. As shown in Figure 8, if user A wants to look for a hospital within 500 meters nearby, the atomic regions covered by the query are $\{\mathrm{AR}_2, \mathrm{AR}_6, \mathrm{AR}_7, \mathrm{AR}_8, \mathrm{AR}_9, \mathrm{AR}_{11}, \mathrm{AR}_{12}, \mathrm{AR}_{13}\}$. The query q generated by the user is $(qv_R, \{\mathrm{AR}_j\}, qv_{attr})$, $(1 \leqslant j \leqslant n$, $AR_j \cap ran(loc, r) \neq \varnothing)$, where r is the query radius specified by user and n is the number of atomic regions contained in the region R. At the end, the user sends (id_i, pk_i, q) to CSP. Algorithm 2 depicts the process of query generation, where qv_R means a vector of region R generated by Bloom Filter and the symbol "|" represents bitwise OR operation over two vectors.

Algorithm 2 $QueryGen(R, attr, \{AR_j\}) \rightarrow (q)$

```
1: for all $h \in h_1$ do
2:     $qv_{R_i} \leftarrow h(k_1, R)$
3:     $qv_R \leftarrow qv_R | qv_{R_i}$
4: end for
5: for all $h \in h_2$ do
6:     $qv_{attr_i} \leftarrow h(k_2, attr)$
7:     $qv_{attr} \leftarrow qv_{attr} | qv_{attr_i}$
8: end for
9: $q \leftarrow (qv_R, qv_{attr}, \{AR_j\})$
10: return $q$
```

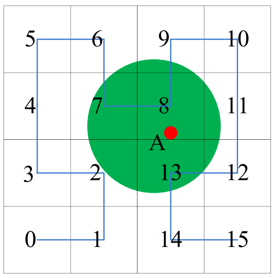


FIGURE 8. Query range of user A.

The location of the query is cloaked by region to prevent disclosure. The location and attribute of the query are hidden by the key-hash functions, which makes that although CSP can obtain the Bloom filter value of the location and attribute, the specific information of the location and the attribute can not be obtained.

D. QUERY PROCESSING

After receiving the user's query request (id_i, pk_i, q), CSP forwards id_i, pk_i and qv_R included in the user's query q to LBSP. Then, LBSP invokes the *Search1* algorithm to find the atomic region data of the region R. In the *Search1* algorithm, the *MatchTest1* algorithm is called, which is used to detect whether the qv_R matches the location index, where t denotes the trapdoor. In this paper, we set $t = k$, which is the number of hash functions in the hash function group. LBSP matches the qv_R with the location index, and finds the bottom node that satisfies trapdoor t in the top-down way. Finally, it will return all the data of the region. The data is stored according to the atomic region, and the data in the same atomic region forms an attribute index. LBSP replies the atomic region data to the CSP, and generates re-encryption key $\mathrm{rk}_{\mathrm{p}\rightarrow i}$ using its own private key $\mathrm{sk_p}$, CSP's public key $\mathrm{pk_{pro}}$, and the user's public key pk_i. At last, LBSP sends re-encryption key $\mathrm{rk}_{\mathrm{p}\rightarrow i}$

Algorithm 3 $Search1(id_i, \mathrm{pk}_i, \mathrm{pk}_{\mathrm{pro}}, \mathrm{sk}_{\mathrm{p}}, qv_R, C_{DB}) \rightarrow (C_{res}, rk_{\mathrm{p}\rightarrow i})$

```
1: if $verify(id_i, \mathrm{pk}_i) = True$ then
2:     if $MatchTest1(qv_R, LT) = True$ then
3:         $rk_{\mathrm{p}\rightarrow i} \leftarrow rkGen(\mathrm{sk}_{\mathrm{p}}, \mathrm{pk}_i, \mathrm{pk}_{\mathrm{pro}})$
4:         return $(C_{res}, rk_{\mathrm{p}\rightarrow i})$
5:     end if
6: end if
7: return null
```

Algorithm 4 $MatchTest1(qv, LT) \rightarrow True/False$

```
1: extract $root$ from $LT$
2: if $qv^T \cdot root \geqslant t$ then
3:     add $root$ into a queue
4:     while the queue is not empty do
5:         pop a node $nd$ from queue
6:         for all children $ch$ of $nd$ do
7:             if $qv^T \cdot ch \geqslant t$ then
8:                 add $ch$ into a queue
9:             end if
10:        end for
11:    end while
12:    if $nd$ is a leaf then
13:        extract $\{C_{AR}\}, \{AT\}$ of $nd$
14:        $C_{res} \leftarrow (\{C_{AR}\}, \{AT\})$
15:        return True
16:    end if
17: else
18:    return False
19: end if
```

to CSP. We will describe *Search1* algorithm and *MatchTest1* algorithm in algorithm 3 and 4 respectively. In algorithm 3, the *verify* algorithm is used to verify the legitimation of user, and the *rkGen* algorithm means the *ReKeyGen* algorithm in APRE, the implementation process of which will be introduced in the re-encryption phase. The input LT of algorithm 4 represents location hierarchical index tree, C_{AR} and AT denote the encrypted data and attribute hierarchical index tree in atomic region respectively.

After receiving the data sent by LBSP, CSP calls the *Search2* algorithm to find the result data for each atomic region of the user query. In this algorithm, *MatchTest2* algorithm is called to detect whether the query attribute qv_{attr} matches the attribute index. *MatchTest2* algorithm is similar to *MatchTest1* algorithm. A top-down approach is used to look up the deepest node that satisfies the trapdoor t, which includes the storage location of the relevant dataset in the atomic region, and the result data set C_{record} is returned. We will describe the *Search2* algorithm and the *MatchTest2* algorithm in algorithms 5 and 6, respectively. The input $\{\mathrm{AR}_j\}$ of algorithm 5 represents the set of atomic region that user requests for.

Algorithm 5 $Search2(\{AR_j\}, qv_{attr}, C_{res}) \rightarrow C_{record}$

```
1: extract $\{C_{AR}\}, \{AT\}$ from $C_{res}$
2: for all $AR$ in $R$ do
3:     if $AR \in \{AR_j\}$ then
4:         if $MatchTest2(qv_{attr}, AT, C_{AR}) = True$ then
5:             return $C_{record}$
6:         end if
7:     end if
8: end for
```

Algorithm 6 $MatchTest2(qv_{attr}, AT, C_{AR}) \rightarrow$ True/False

```
1: extract $root$ from $AT$
2: if $qv_{attr}^T \cdot root \geqslant t$ then
3:     add $root$ into the queue
4:     while the queue is not empty do
5:         pop a node $nd$ from the queue
6:         for all children $ch$ of $nd$ do
7:             if $qv_{attr}^T \cdot ch \geqslant t$ then
8:                 add $ch$ into a queue
9:             end if
10:        end for
11:    end while
12:    extract $\{record_i\}$ from $nd$
13:    $C_{record} \leftarrow C_{record} \cup C_{record_i}$
14:    return True
15: else
16:    return False
17: end if
```

Note that during query processing, the information of user's location and attribute is preserved. LBSP obtains the user's query area R without knowing its specific location and query attributes, which protects user's location privacy and query privacy from the point of view of region cloak. At the same time, although CSP obtains the query attribute and the atomic region's numbers transformed by the Hilbert curve, it is still difficult for CSP to obtain specific query information due to that CSP doesn't know the key of hash function group $h_2()$ and the generation rule of Hilbert curve.

E. PROXY RE-ENCRYPTION

CSP takes the result data set C_{record}, the proxy's private key $\mathrm{sk}_{\mathrm{pro}}$, and the re-encryption key $\mathrm{rk}_{\mathrm{p}\rightarrow i}$ as the input of the re-encryption function $PRE(\,)$, and outputs the re-encrypted ciphertext set, which the authorized user can decrypt with his own private key. The concrete re-encryption algorithm is shown in algorithm 7.

F. DECRYPTION

When the authorized user receives the POI data set that re-encrypted by CSP, the user can decrypt the data set with his private key to obtain the plaintext information of POI. The decryption algorithm is described in algorithm 8.

To prevent CSP from colluding and misusing the re-encryption key with the user to obtain data that is not

Algorithm 7 $PRE(rk_{\mathrm{p}\to i}, \mathrm{sk}_{\mathrm{pro}}, C_{record}) \to C'_{record}$

```
1: for all $C_{recordi}$ in $C_{record}$ do
2:     $C_{recordi} \leftarrow ReEnc(rk_{\mathrm{p}\to i}, \mathrm{sk}_{\mathrm{pro}}, C_{recordi})$
3:     $C_{record} \leftarrow C_{record} \cup C'_{recordi}$
4: end for
5: return $C'_{record}$
```

Algorithm 8 $Dec(\mathrm{sk}_i, C'_{record}) \to m_{record}$

```
1: for all $C'_{recordi} in C'_{record}$ do
2:     $m_{recordi} \leftarrow Dec(\mathrm{sk}_i, C'_{recordi})$
3:     $m_{record} \leftarrow m_{record} \cup m_{recordi}$
4: end for
5: return $m_{record}$
```

authorized to the user. In this paper, we adopt accountable proxy re-encryption (APRE) technology, through which authorized users can share encrypted data and the decryption ability of the delegator and proxy can be distinguished. In APRE, the *judge* algorithm is used to judge the creator of the decryption device so as to track the decryption behavior of the delegator and the proxy. Specific description of *judge* algorithm is shown as follows.

Setup(λ). Let λ be a security parameter, G and G_T be groups of prime order p, and $e : G \times G \to G_T$ be a bilinear map. Select g_1, g_2, h_1, h_2 randomly from G and calculate $L = e(h_1, h_2)$. The system parameters are $param = (g_1, g_2, h_1, h_2, L)$.

ProxySet$(param)$. Select $z \leftarrow Z_p$ randomly and uniformly, calculate $Z = g_2^z$. Set $\mathrm{sk}_{\mathrm{pro}} = z$ as the private key and $\mathrm{pk}_{\mathrm{pro}} = Z$ as the public key.

KenGen$(param)$. Select $x_i, y_i \leftarrow Z_p$ randomly and uniformly and calculate $(X_i, Y_i) = (h_1^{x_i}, g_1^{y_i})$. Set $\mathrm{sk}_i = (x_i, y_i)$ as the private key and $\mathrm{pk}_i = (X_i, Y_i)$ as the public key.

ReKeyGen$(\mathrm{sk}_{\mathrm{p}}, \mathrm{pk}_i, \mathrm{pk}_{\mathrm{pro}})$. Taken $\mathrm{sk}_{\mathrm{p}}, \mathrm{pk}_i, \mathrm{pk}_{\mathrm{pro}}$ as input, calculate

$$W = (h_2 Y_j Z)^{1/x_i},$$

and set $\mathrm{rk}_{\mathrm{p}\to i} = W$ as the re-encryption key.

Enc$_1(\mathrm{pk}_i, m)$. Taken pk_i and m as inputs, select $r \leftarrow Z_p$ randomly and uniformly, calculate

$$c'_0 = L^r \cdot m, \quad c'_1 = g_1^r, \quad c'_2 = L^r \cdot e(h_1, Y_i)^r.$$

Set $C'_i = (c'_0, c'_1, c'_2)$ as the second layer of ciphertext.

Enc$_2(\mathrm{pk}_{\mathrm{p}}, m)$. Taken pk_{p} and m as inputs, select $r \leftarrow Z_p$ randomly and uniformly, calculate

$$c_0 = L^r \cdot m, c_1 = g_1^r, \quad c_2 = e(h_1, g_2)^r, \quad c_3 = X_{\mathrm{p}}^r.$$

Set $C_{\mathrm{p}} = (c_0, c_1, c_2, c_3)$ as the second layer of ciphertext.

ReEnc$(\mathrm{rk}_{\mathrm{p}\to i}, \mathrm{sk}_{\mathrm{pro}}, C_{\mathrm{p}})$. Given $\mathrm{rk}_{\mathrm{p}\to i}, \mathrm{sk}_{\mathrm{pro}}, C_{\mathrm{p}}$ as inputs, calculate

$$c'_0 = c_0, c'_1 = c_1, \quad c'_2 = e(c_3, W)/c_2^z.$$

Set $C'_i = (c'_0, c'_1, c'_2)$ as a re-encrypted ciphertext.

Dec$_1(\mathrm{sk}_i, C'_i)$. Taken sk_i and C'_i as inputs, calculate the message

$$m = c'_0 \cdot e(h_1, c'_1)^{y_i}/c'_2.$$

Dec$_2(\mathrm{sk}_{\mathrm{p}}, C_{\mathrm{p}})$. Taken sk_{p} and C_{p} as inputs, calculate the message

$$m = c_0/e(c_3, h_2)^{1/x_{\mathrm{p}}}.$$

Judge$^{D_{\mathrm{p},u}}(\mathrm{pk}_{\mathrm{p}},\mathrm{pk}_{\mathrm{pro}})$. Taken the decryption device $D_{\mathrm{p},u}$ as oracle and $\mathrm{pk}_{\mathrm{p}}, \mathrm{pk}_{\mathrm{pro}}$ as the inputs, the algorithm is as follows.

1) Repeat the following experiment $n = \lambda/\mu$ times.
   - Choose r, $r' \in Z_p$ and $m \in G_T$ uniformly and randomly, compute
$$C = (c_0 = m \cdot L^r \cdot e(h_1, Z)^{r-r'}, c_1 = g_1^r, c_2 = e(h_1, g_2)^{r'}, c_3 = X_{\mathrm{p}}^r).$$
   - Take C as the input of decryption device $D_{\mathrm{p},u}$ and obtain the output m'.
   - If $m = m'$, output "Proxy" and exit.
2) Otherwise, output "Delegator".

VI. SECURITY ANALYSIS

In this section, we first define the leak function, which represents all the information that the adversary is allowed to capture during the query process. Then, we define the data privacy and query privacy under the selected plaintext attack model. Finally, we prove that our plan meets the security goals.

A. LEAKAGE FUNCTION

The leak function represents all the information that the opponent can obtain throughout the system. It includes the query area qv_R sent by the user to the LBSP, region division $\{\mathrm{AR}_i\}(1 \leqslant i \leqslant n)$ sent by LBSP to users, query q sent by user to the CSP, the encrypted data, attribute index qv_{attr} sent by LBSP to CSP and so on. According to the proposed concept, the leakage function consists of search pattern, access pattern and size pattern, which will be described separately.

Definition 1: Size Pattern τ: Let $C_{record} = \{C_{record_1}, \ldots, C_{record_m}\}$, $Hilbert(R) = \{\mathrm{AR}_1, \ldots, \mathrm{AR}_n\}$, $\mathrm{AT} = \{\mathrm{AT}_1, \ldots, \mathrm{AT}_n\}$, and q denote the encrypted dataset in one atomic region, the atomic region set of region R encoded by Hilbert curve, encrypted attribute index set in region R and query q, where m and n represents the total number of POI records in one atomic region and the total number of atomic regions contained in region R respectively. That is, for each atomic region AR_i, it generates an encrypted dataset C_{record} and attribute index AT_i. Size Pattern $\tau = \{|C_{record}|, |Hilbert(R)|, |\mathrm{AT}|, |q|\}$.

Definition 2: Search Pattern ζ: Let $\{t_1, t_2, \ldots, t_q\}$ represents q consecutive queries. For each t_i, it represents a tuple composed of the region qv_R, atomic domain set $\{\mathrm{AR}_i\}$, the attribute qv_{attr}, and $\{C_1, C_2, \ldots, C_q\}$ represents the corresponding retrieval record. If the query region qv_R, atomic region set $\{AR_i\}$ and the attribute qv_{attr} appear at the same time in the retrieved C_i, then $\zeta = 1$, otherwise $\zeta = 0$.

Definition 3: Access Pattern ς: let C_{IV} be an encrypted index, which includes $Hilbert(R)$ and AT. $\{t_1, t_2, \ldots, t_q\}$ represents q queries, and $\{C_1, \ldots, C_q\}$ represents the corresponding retrieval results. So the access mode is $\{C_{IV}(t_1), C_1, \ldots, C_{IV}(t_q), C_q\}$.

Definition 4: Leakage Function L: let C_{IV}, C_{record}, q represent encrypted index, encrypted dataset and query respectively. The leakage function is represented as $L = \{C_{IV}, C_{record}, q, \tau, \zeta, \varsigma\}$.

B. SECURITY DEFINITION

Before giving the security definition of the proposed scheme, we first introduce the definition of ciphertext-indistinguishability under a chosen plaintext attack.

Definition 5: A public key encryption scheme $\pi = (Gen, Enc, Dec)$ has ciphertext-indistinguishability under a chosen-plaintext attack if for all probabilistic polynomial time(PPT) adversary A there exists a negligible function $negl$ such that $Pr[Pub^{cpa}_{A,\pi} = 1] \leqslant \frac{1}{2} + negl(\lambda)$.

Based on the above disclosure analysis, we define the security of our scheme from two aspects: data privacy and query privacy.

Definition 6: DBDH Assumption. For an algorithm B, we define its advantage as

$$Adv_B^{DBDH}(\lambda) = |Pr[B(g, g^a, g^b, g^c, e(g,g)^{abc}) = 1] - Pr[B(g, g^a, g^b, g^c, e(g,g)^t) = 1]|$$

where a, b, c, t are randomly selected from Z_p. We say DBDH (Decisional Bilinear Diffie-Hellman) assumption holds, if for any probabilistic polynomial time (PPT) algorithm B, its advantage is negligible in λ.

1) DATA PRIVACY

We define data privacy by first uploading two databases DB_0 and DB_1 to the challenger, and allow the adversary to send the adaptive queries based on the leakage analysis before making a final decision about which database is being used. The formal definition is as follows.

Let Π =($Setup, Init, DBEnc, QueryGen, Search1, Search2$) be a privacy-preserving outsourcing LBS scheme. For a PPT adversary A, the advantage function $ADV_A^{\pi}(1^{\lambda})$ is defined as $ADV_A^{\pi}(1^{\lambda}) = Pr(b^* = b) - \frac{1}{2}$, where b^* and b are generated as follows.

Init: The adversary submits two databases DB_0 and DB_1 with the same number of records and index structure to the challenger.

Setup: The challenger runs $Setup(1^{\lambda})$ to generate the required parameters and keys.

Phase 1: The adversary properly submits requests in one of the following two forms:

(1) Ciphertext request: the adversary submits a database $DB_j (j = 0, 1)$ and requests an encrypted record $C_i (1 \leqslant i \leqslant |DB_j|)$.

(2) Query request: The adversary sends a query Q, which satisfies the equation that $L(Q, C_{IV_0}, C_{DB_0}) = L(Q, C_{IV_1}, C_{DB_1})$.

Challenge: The challenger randomly selects a bit b from $\{0, 1\}$, and then generates index C_{IV_b} and the encrypted database C_{DB_b} by the *DBEnc* algorithm.

Phase2: Like the constraints in phase1, the adversary continues to adaptively send queries to challenger.

Guess: The adversary outputs a bit b' as a guess of b.

The data privacy of scheme π can be held under chosen plaintext model if for any PPT adversary A, the advantage function is a negligible function in λ.

2) QUERY PRIVACY

The query privacy is similar to the data privacy, and the difference is that query privacy submits two queries rather than two databases. The details are as follows.

Let Π = (*Setup,Init,DBEnc,QueryGen,Search1,Search2*) be a privacy-preserving outsourcing LBS scheme. For a PPT adversary A, the advantage function $ADV_A^{\pi}(1^{\lambda})$ is defined as $ADV_A^{\pi}(1^{\lambda}) = Pr(b^* = b) - \frac{1}{2}$, where b^* and b are generated in following way.

Init: The adversary submits two initial queries q_0 and q_1 to the challenger.

Setup: The challenger runs $Setup(1^{\lambda})$ to generate the required parameters and keys.

Phase 1: The adversary A properly submits a request in one of the following two forms:

(1) Ciphertext request: the adversary submits a database DB to the challenger and requests an encrypted record C_i $(1 \leqslant i \leqslant |DB|)$. The challenger responds C_i, if $L(q_0, C_{IV}, C_{DB}) = L(q_1, C_{IV}, C_{DB})$.

(2) Query request: The adversary sends the original query q to the challenger, and the challenger runs the *QueryGen* algorithm to generate the query Q and sends it to the adversary.

Challenge: The challenger randomly selects a bit b from $\{0, 1\}$, and q_b is used as the input of the *QueryGen* algorithm, then outputs Q_b.

Phase2: Like the constraints in phase1, the adversary continues to adaptively send queries to the challenger.

Guess: The adversary outputs a bit b' as a guess of b.

The query privacy of scheme π can be held under chosen plaintext model if for any PPT adversary A, the advantage function is a negligible function in λ.

C. SECURITY PROOF

In this paper, the Bloom Filter, the Hilbert filling curve and the proxy re-encryption technique are embedded into the index generation and the query generation, the region division, and the data re-encryption stage, respectively. By using these mature tools, our scheme can achieve privacy preservation. We will consider its security separately from the perspective of construction tools.

Theorem 1: Hilbert spatial transformation is unidirectional. Without knowing the spatial transformation parameters, it is difficult to determine the position in the two-dimensional space based on the one-dimensional Hilbert value.

Given the order N of Hilbert curve, the starting unit $< x_0, y_0 >$, the curve direction Γ, and the scale factor Θ, a Hilbert space filling curve can be uniquely identified.

For a malicious adversary, it is not feasible to determine spatial transformation parameters by comparing Hilbert values for all original points.

Brute-force attack: Let the starting point of the curve be $< x_0, y_0 >$, and m be the number of bits of x_0 and y_0. In order to obtain the starting point of the curve, the malicious adversary needs to generate 2^m values on the X-axis and Y-axis respectively. Similarly, assume that the direction of the curve expressed with m bits, the full 360° Continuous space can be discrete into 2^m values. The adversary must try 2^m values to determine the direction of the curve. Given the scale factor, let N be the order of Hilbert curve, then there are $(N \cdot 2^m \cdot 2^m \cdot 2^m)$ choices in the whole space. Therefore, the complexity of obtaining spatial transformation parameters through brute-force attack is (2^{3q}). Given a large enough value of m, this makes the Hilbert transformation irreversible.

Theorem 2: The accountable proxy re-encryption (APRE) technology used in this paper can realize the CPA security at the second level ciphertexts and the first level ciphertexts under the DBDH assumption.

Lemma 1: The APRE scheme is CPA secure at the second level ciphertext under the DBDH assumption.

Proof: Let A be a CPA adversary, which attacks the second level ciphertext with the advantage of ϵ. We construct an algorithm B to solve the DBDH problem by interacting with adversary A. Algorithm B takes a random challenge (g, g^a, g^b, g^c, T) as input, and its goal is to determine whether $T = e(g, g)^{abc}$ or $T = e(g, g)^t$, where $a, b, c, t \leftarrow Z_p$ are chosen randomly and uniformly.

Setup(λ). Choose $\delta \leftarrow Z_p$ uniformly at random, set $g_1 = g$, $g_2 = g^{\delta}$, $h_1 = g^a$, $h_2 = g^b$, $L = e(g^a, g^b)$. Return *param* $= (g_1, g_2, h_1, h_2, L)$ as the system parameter.

ProxySet(*param*). Choose $z \leftarrow Z_p$ uniformly at random and compute $Z = g_2^z$. Let the proxy's private key be $\mathrm{sk}_{\mathrm{pro}} = z$ and public key be $\mathrm{pk}_{\mathrm{pro}} = Z$. Return $\mathrm{pk}_{\mathrm{pro}}$.

*Phase*1. B responds A's query as follows.

- $O_{pyr}(\lambda)$. Return the proxy's private key $\mathrm{sk}_{\mathrm{pro}} = z$.
- $O_{hkg}(i)$. Choose $x_i, y_i \leftarrow Z_p$ uniformly at random.
  - If B guesses the k-th key generation query will be the target user. Let $i^* = i$, calculate
  $$\mathrm{pk}_{i^*} = (X_i = g^{x_{i^*}}, Y_i = g^{y_{i^*}}),$$
  and implicitly set $\mathrm{sk}_i^* = (x_{i^*}/a, y_{i^*})$. Return pk_{i^*}.
  - otherwise, calculate
  $$\mathrm{pk}_i = (X_i = h_1^{x_i}, Y_i = h_2^{-1} g^{y_i}),$$
  and implicitly set $\mathrm{sk}_i = (x_i, y_i - b)$. Return pk_i.
- $O_{ckg}(i)$. Choose $x_i, y_i \leftarrow Z_p$ uniformly at random, set $\mathrm{sk}_i = (x_i, y_i)$. Calculate
  $$\mathrm{pk}_i = (X_i = h_1^{x_i}, Y_i = g^{y_i}),$$
  and return $(\mathrm{sk}_i,\mathrm{pk}_i)$.
- $O_{rkg}(\mathrm{pk}_i, \mathrm{pk}_j, \mathrm{pk}_{pro})$. Given $(\mathrm{pk}_i, \mathrm{pk}_j)$, where pk_i and pk_j are produced by O_{hkg} or O_{ckg}, B distinguishes the following cases.
  - If $\mathrm{pk}_i \neq \mathrm{pk}_i^*$, run *ReKeyGen*$(\mathrm{sk}_i, \mathrm{pk}_j, \mathrm{pk}_{\mathrm{pro}})$ and return its result.
  - If $\mathrm{pk}_i = \mathrm{pk}_{i^*}$, and pk_j is uncorrupted, calculate
  $$W = (h_2 Y_j Z)^{\left(\frac{x_{i^*}}{a}\right)^{-1}} = (g^a)^{\frac{y_j + z \cdot \delta}{x_{i^*}}}.$$
  Return $\mathrm{rk}_{i \to j} = W$.
  - If $\mathrm{pk}_i = \mathrm{pk}_i^*$, and pk_j is corrupted. Return $\perp$.

Challenge. When the *phase*1 is finished,the adversary A outputs m_0, m_1 and a target public key pk*. If $\mathrm{pk}_{i^*} \neq$pk*, B outputs a random bit and discards. Otherwise, B selects a random bit $b \in \{0, 1\}$ and calculates

$$c_0 = T \cdot m_b, \quad c_1 = g_1^r = g^c,$$
$$c_2 = e(h_1, g_2)^r = e(g^a, g^c)^{\delta}, \quad c_3 = X_i^r = (g^c)^{x_{i^*}},$$

implicitly setting $r = c$. If adversary A makes O_{hkg} queries at most q_{hkg} times. Then, the probability that B guesses the correct i^* in challenge stage is at least $1/q_{hkg}$.

*Phase*2. Like *Phase*1, B responds A's query. Finally, A outputs a guess bit $b' \in \{0, 1\}$. If $b = b'$, B outputs 1, otherwise outputs 0. The advantage that B solves the DBDH problem is at least $\frac{1}{q_{hkg}} \cdot \epsilon$. End.

Lemma 2: The APRE scheme is CPA secure at the first level ciphertext under the DBDH assumption.

Proof let A be a CPA adversary, which attacks the first level ciphertext with the advantage of ϵ. We construct an algorithm B to solve the DBDH problem by interacting with adversary A. Algorithm B takes a random challenge (g, g^a, g^b, g^c, T) as input, and B's goal is to determine whether $T = e(g, g)^{abc}$ or $T = e(g, g)^t$, where $a, b, c, t \leftarrow Z_p$ are chosen randomly and uniformly.

Setup(λ). Choose $\delta \leftarrow Z_p$ uniformly at random, and set $g_1 = g$, $g_2 = g^{\delta}$, $h_1 = g^a$, $h_2 = g^b$, $L = e(g^a, g^b)$. Return *param* $= (g_1, g_2, h_1, h_2, L)$ as the system parameter.

ProxySet(*param*). Choose $z \leftarrow Z_p$ uniformly at random and compute $Z = g_2^z$. Let the proxy's private key be $\mathrm{sk}_{\mathrm{pro}} = z$ and public key be $\mathrm{pk}_{\mathrm{pro}} = Z$. Return $\mathrm{pk}_{\mathrm{pro}}$.

*Phase*1. B responds A's query as follows.

- $O_{pyr}(\lambda)$. Return the proxy's private key $\mathrm{sk}_{\mathrm{pro}} = z$.
- $O_{hkg}(i)$. Choose $x_i, y_i \leftarrow Z_p$ uniformly at random.
  - If B guesses the k-th key generation query will be the target user. Let $i^* = i$, calculate
  $$\mathrm{pk}_{i^*} = (X_i = h_1^{x_{i^*}}, Y_i = h_2^{-1} g^{y_{i^*}}),$$
  and implicitly set $\mathrm{sk}_i^* = (x_{i^*}, y_{i^*} - b)$. Return pk_{i^*}.

- otherwise, calculate

$$\text{pk}_i = (X_i = h_1^{x_i}, Y_i = g^{y_i}),$$

and implicitly set $\text{sk}_i = (x_i, y_i)$. Return pk_i.

- $O_{ckg}(i)$. Choose $x_i, y_i \leftarrow Z_p$ uniformly at random, set $\text{sk}_i = (x_i, y_i)$. Calculate

$$\text{pk}_i = (X_i = h_1^{x_i}, Y_i = g^{y_i}),$$

and return $(\text{sk}_i, \text{pk}_i)$.
- $O_{rkg}(\text{pk}_i, \text{pk}_j)$. Given $(\text{pk}_i, \text{pk}_j)$, where pk_i and pk_j are produced by O_{hkg} or O_{ckg}. B runs $ReKeyGen(\text{sk}_i, \text{pk}_j)$ and returns its result.

Challenge. When the *phase*1 is finished, the adversary A outputs m_0, m_1 and a target public key pk*. If $\text{pk}_{i^*} \neq \text{pk}^*$, B outputs a random bit and discards. Otherwise, B selects a random bit $b \in \{0, 1\}$ and calculates

$$c_0' = T \cdot m_b, \quad c_1' = g_1^r = g^c,$$
$$c_2' = L^r \cdot e(h_1^r, Y_{i^*}) = e(g^a, g^c)^{y_{i^*}},$$

implicitly setting $r = c$. If adversary A makes O_{hkg} queries at most q_{hkg} times. Then, the probability that B guesses the correct i^* in challenge stage is at least $1/q_{hkg}$.

*Phase*2. Like *Phase*1, B responds A's query. Finally, A outputs a guess bit $b' \in \{0, 1\}$. If $b = b'$, B outputs 1, otherwise outputs 0. The advantage that B solves the DBDH problem is at least $\frac{1}{q_{hkg}} \cdot \epsilon$. End.

Theorem 3: The scheme can achieve privacy preservation against untrusted LBSP.

Proof: In the scheme, LBSP can receive message about user's identity and query area qv_R from CSP. LBSP cannot know the exact location and query content of user based on above information in addition to random guesses. Therefore, the scheme can preserve user's location privacy and query privacy against untrustworthy LBSP.

Theorem 4: The scheme can realize privacy preservation against untrusted CSP.

Proof: On the one hand, CSP obtains the user's identity id_i and query q, which includes region qv_R, atomic region set $\{AR_i\}$, and query attribute qv_{attr}. Due to the unidirectionality of key-hash functions and Hilbert space transformation, CSP cannot know the plaintext information about query region R and atomic region, and thus cannot know the user's exact location. Similarly, the CSP cannot obtain the user's query content. On the other hand, CSP gets encrypted POI data from LBSP. Due to the discrete logarithm problem based on proxy re-encryption technology, CSP cannot know plaintext POI data, so as to ensure data privacy against untrusted CSP.

VII. PERFORMANCE ANALYSIS

We analyze the performance of the scheme according to the computational and communication cost on the user, CSP, LBSP.

A. COMPUTATIONAL COST

Firstly, we analyze the computational cost on the user. In the scheme, the user performs step 1 (query generation) and step 6 (decryption). The processing time of query generation is mainly spent on generating Bloom Filter value of query area and query attribute, and its computational complexity is $O(k \cdot m)$, where k is the number of hash functions in Bloom Filter and m is the bit length of Bloom Filter. In step 6, the user needs to decrypt the re-encrypted POI data set, and its computational complexity is related to the amount of POI in the query region. The larger the order of Hilbert curve is, the smaller the atomic region is and the less POI data it contains. The amount of POI data in the query area is inversely proportional to N^2, but increases as the total of all POIs n increases. Therefore, the computional cost of the user is $O(km + n \cdot N^{-2})$.

Then, we analyze the processing time of CSP. In the scheme, CSP performs step 2 (query forwarding), step 4 (attribute matching), and step 5 (proxy re-encryption). The computional complexity of step 2 is negligible because it only needs to perform a forwarding operation. In step 4, the CSP matches the user query attribute with the attribute index, which has a computational complexity of $O(\log(n \cdot N^{-2}))$. In step 5, CSP re-encrypts the POI that satisfy the query attribute, the computational overhead of which is related to the amount of POI in the atomic region, and its computational complexity is $O(n \cdot N^{-2})$. Hence, the computational cost of CSP is $O(\log(n \cdot N^{-2}) + n \cdot N^{-2})$.

Finally, we analyze the processing time of LBSP. In the process of generating query, the user needs to request atomic region coding from LBSP. At this time, LBSP will use Hilbert curve to divide the region into atomic regions, the computational complexity of which is related to the order N of Hilbert curve. Furthermore, LBSP will perform step 3 (POI encryption). Before the encryption operation is performed, the query region needs to be matched to the location index. The computional cost of preprocessing operation is a fixed value that can be ignored. The processing time of step 3 is related to the amount of POI contained in the query region, and its computational time is $O(n + N^2)$.

B. COMMUNICATION TIME

Firstly, we analyse the communication overheads between the user and CSP. The message that the user outputs is a LBS query with a fixed size. The message that the user receives from the CSP is POI data that satisfies the query attribute in the specified atomic region, and the communication cost of which is $O(n \cdot N^{-2})$. We then analyze the communication cost between CSP and LBSP. The transfer-out message of CSP is the user's query request, The transfer-in message of CSP is the encrypted data set, whose communication cost is given by $O(n)$.

We describe the meaning of notation involved in the performance analysis in Table 2 and generalize the computational and communication overheads of all entities in Table 3.

TABLE 2. Description of notation.

Notation	Description
m	Bit length of Bloom Filter
k	Number of hash functions
n	Number of total POI in a region
N	Order of Hilbert curve

TABLE 3. Performance analysis.

Entity	Computation cost	Communication cost
user	$O(k \cdot m + n \cdot N^{-2})$	$O(n \cdot N^{-2})$
CSP	$O(\log(n \cdot N^{-2}) + n \cdot N^{-2})$	$O(n \cdot N^{-2} + n)$
LBSP	$O(n + N^2)$	$O(n)$

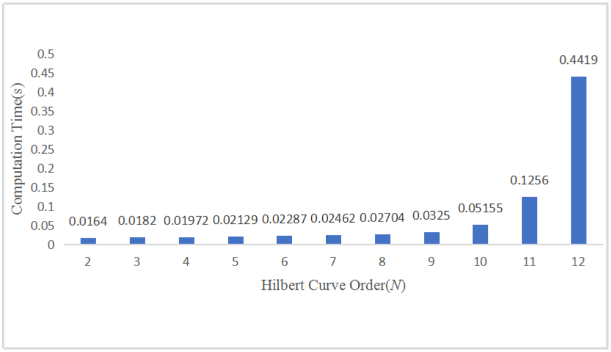


FIGURE 9. Hilbert Curve Order.

VIII. EVALUATION

We use the existing tools such as Hilbert curve, Bloom Filter and accountable proxy re-encryption in the construction of the scheme. In this part, we will analyse the construction effectiveness of the above tools, which can indirectly prove the effectiveness of the scheme. The experiments were performed on a 64-bit local PC with an Intel Core i5-3230m processor at 2.6 GHz and 8 GB RAM. MATLAB R2014a was used to implement the construction of Hilbert Curve.Meanwhile, the Java Development Kit (JDK) and Eclipse Integrated Development Environment (IDE) were used to implement Bloom Filter class.

A. HILBERT CURVE

In this paper, Hilbert Curve is used to divide region into atomic regions which realizes fine-grained location query. The order of Hilbert Curve has an effect on partition efficiency as well as query efficiency. The higher the order of Hilbert curve, the more atomic regions there are, and the less data of POIs there are in each atomic region. Figure 9 describes the computation time required to construct Hilbert Curve with different orders. It can be seen from the picture that the construction of Hilbert Curve takes more time as the order increases, but this time is relatively small compared to the whole scheme. When $N = 12$, it only takes 0.44ms to construct Hilbert Curve. Therefore, it is efficient to use Hilbert Curve to divide regions.

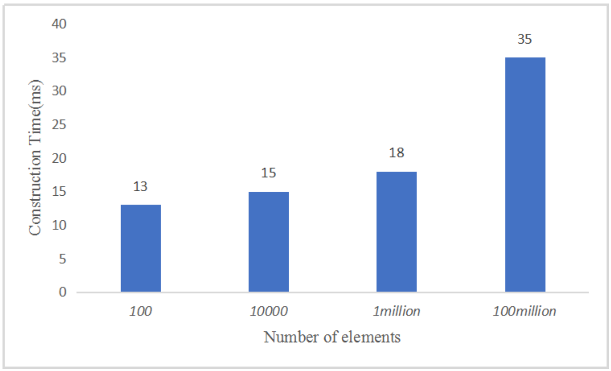


FIGURE 10. Number of elements in Bloom Filter.

TABLE 4. Evaluation of APRE.

Key and Ciphertext Size		Computational Cost	
Name	Size	Stage	Computational Cost
pk	$2\|e_G\|$	Enc_1	$t_p + t_e + 2t_e^{'}$
rk	$\|e_G\|$	Enc_2	$t_p + 2t_e + 2t_e^{'}$
C_1	$2\|e_{G_T}\| + \|e_G\|$	$ReEnc$	$t_p + t_e^{'}$
C_2	$2\|e_{G_T}\| + 2\|e_G\|$	Dec_1	$t_p + t_e^{'}$
$C^{'}$	$2\|e_{G_T}\| + \|e_G\|$	Dec_2	$t_p + t_e^{'}$

B. BLOOM FILTER

Bloom Filter is used to generate location index and attribute index. In the construction of Bloom Filter, the user only needs to input the element number n and the expected error rate p, and the implementation algorithm will automatically generate the Bloom Filter with the optimal length m and the optimal numbers of hash function k. Here, we set the error rate $p = 0.1$ as default. Comparing the effects of different magnitude of input elements on construction time of Bloom Filter, it can be seen that the construction time is maintained within 20 ms with the input elements numbering 100-1000000. Only when the input element reaches more than 100 million, it takes 35ms to construct Bloom Filter. Therefore, it is effective to use Bloom Filter to construct location index for real datasets less than 100 million. Figure 10 describes the impact of magnitude of input elements on the construction time of Bloom Filter.

C. APRE

In our scheme, we use accountable proxy re-encryption technology to realize the sharing of encrypted POIs among different authorized users. Based on the construction of proxy re-encryption in literature [45], we will describe the efficiency of proxy re-encryption in detail next. In table 4, $|e_G|$ represents the size of elements in group G and $|e_{G_T}|$ represents the size of elements in group G_T. t_p, t_e, $t_e^{'}$ represent the time required for one bilinear mapping, one exponential operation in group G, one exponential operatioon in group G_T. Table 4 describes the key and ciphertext size involved in proxy re-encryption and some operations involved in encryption and decryption. Figure 11 depicts the time that accountable proxy re-encryption takes at different phases.

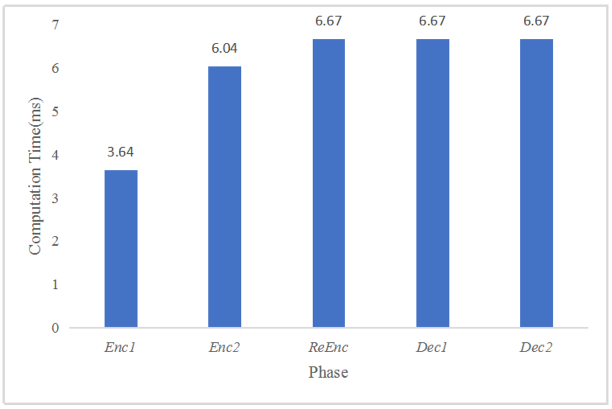


FIGURE 11. Computation Cost of APRE.

IX. CONCLUSION

In this paper, we propose an outsourced LBS privacy protection scheme. Bloom Filter, Hilbert curve, APRE are used to realize the process of query generation, partition of region, sharing of encrypted data and so on. Thus, The authorization access of encrypted data in outsourced environment is completed.

In order to solve the problem of peripheral recommendation, the user customizes the query range, and the atomic region sets encoded by Hilbert curve are used to represent the user's query range. Obviously, there is some inaccuracy in this representation. In the following work, we will look for more accurate range query methods.

In addition, we will extend our work to support k-nearest neighbor queries with location-sensitive hash functions, where the hash interpolation between the query and the POI represents the distance between them. By finding k hash values nearest to the query location, k nearest neighbors are found, thus k-nearest neighbor (KNN) query can be realized.

REFERENCES

[1] B. Lee, J. Oh, H. Yu, and J. Kim, "Protecting location privacy using location semantics," in *Proc. ACM SIGKDD*, New York, NY, USA, 2011, pp. 1289–1297.

[2] IETF. (2012). *Geographic Location/Privacy Working Group*. [Online]. Available: http://datatracker.ietf.org/wg/geopriv/charter/

[3] W3C. (2012). *Platform for Privacy Preferences (P3P) Project*. [Online]. Available: http://www.w3.org/P3P

[4] M. L. Yiu, C. S. Jensen, X. Huang, and H. Lu, "SpaceTwist: Managing the trade-offs among location privacy, query performance, and query accuracy in mobile services," in *Proc. ICDE*, Piscataway, NJ, USA, Apr. 2008, pp. 366–375.

[5] M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," in *Proc. Mobisys*, San Francisco, CA, USA, 2003, pp. 31–42.

[6] M. F. Mokbel, C.-Y. Chow, and W. G. Aref, "The new Casper: Query processing for location services without compromising privacy," in *Proc. VLDB*, Seoul, South Korea, 2006, pp. 763–774.

[7] B. Gedik and L. Liu, "Protecting location privacy with personalized k-anonymity: Architecture and algorithms," *IEEE Trans. Mobile Comput.*, vol. 7, no. 1, pp. 1–18, Jan. 2008. doi: 10.1109/TMC.2007.1062.

[8] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias, "Preventing location-based identity inference in anonymous spatial queries," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 12, pp. 1719–1733, Dec. 2007. doi: 10.1109/TKDE.2007.190662.

[9] Z. Huo, X. Meng, H. Hu, and Y. Huang, "You can walk alone: Trajectory privacy-preserving through significant stays protection," in *Proc. DASFAA*, Busan, South Korea, 2012, pp. 351–366.

[10] S. Gao, J. Ma, C. Sun, and X. Li, "Balancing trajectory privacy and data utility using a personalized anonymization model," *J. Netw. Comput. Appl.*, vol. 38, pp. 125–134, Feb. 2014. doi: 10.1016/j.jnca.2013.03.010.

[11] Y. Wu, Q. Tang, W. Ni, Z. Sun, and S. Liao, "A clustering hybrid basedalgorithm for privacy preserving trajectory data publishing," (in Chinese), *J. Comput. Res. Develop.*, vol. 50, no. 3, pp. 578–593, Apr. 2013.

[12] J. Li, Z.-H. Bai, R.-Y. Yu, Y.-M. Cui, and X.-W. Wang, "Mobile location privacy protection algorithm based on PSO optimization," (in Chinese), *Chin. J. Comput.*, vol. 41, no. 5, pp. 1037–1051, Jul. 2017. doi: 10.11897/SP.J.1016.2018.01037.

[13] C. L. Li, X. Lv, and X. Li, "Dynamic K-anonymity algorithm for resisting prediction attack based on historical trajectories," (in Chinese), *Comput. Eng. Appl.*, vol. 54, no. 2, pp. 119–124, Jan. 2018. doi: 10.3778/j.issn.1002-8331.1707-0325.

[14] D. W. Lin and Y. F. Wang, "False trajectory generating method based on user's true trajectory," (in Chinese), *Comput. Eng.*, vol. 44, no. 8, pp. 142–150, Nov. 2018. doi: 10.19678/j.issn.1000-3428.0049930.

[15] C. Dwork, "Differential privacy," in *Proc. ICALP*, Venice, Italy, 2006, pp. 1–12.

[16] R. Assam, M. Hassani, and T. Seidl, "Differential private trajectory protection of moving objects," in *Proc. SIGSPATIAL-IWGS*, New York, NY, USA, 2012, pp. 68–77.

[17] R. Chen, B. C. M. Fung, B. C. Desai, and N. M. Sossou, "Differentially private transit data publication: A case study on the montreal transportation system," in *Proc. ACM SIGKDD*, New York, NY, USA, 2012, pp. 213–221.

[18] P. Xiong, T. Zhu, L. Pan, W. Niu, and G. Li, "Privacy preserving in location data release: A differential privacy approach," in *Proc. PRICAI*, Berlin, Germany, 2014, pp. 183–195.

[19] L. Wang and X.-F. Meng, "Location privacy preservation in big data era: A survey," (in Chinese), *J. Softw.*, vol. 25, no. 4, pp. 693–712, Jun. 2014. doi: 10.13328/j.cnki.jos.004551.

[20] L. Zhang, Y. Liu, and R. C. Wang, "Location publishing technology based on differential privacy-preserving for big data services," (in Chinese), *J. Commun.*, vol. 37, no. 9, pp. 46–54, Nov. 2016.

[21] X.-D. Bi, Y. Liang, H.-Z. Shi, and H. Tian, "Aparameterized location privacy protection method based on two-level anonymity," (in Chinese), *J. Shandong Univ. Natural Sci.*, vol. 52, no. 5, pp. 75–84, Jul. 2017

[22] Y. F. Ma and L. Zhang, "LBS group nearest neighbor query method based on differential privacy," (in Chinese), *Comput. Sci.*, vol. 44, no. Z6, pp. 336–341, Jun. 2017. doi: 10.11896/j.issn.1002-137X.2017.6A.077.

[23] A. R. Beresford and F. Stajano, "Location privacy in pervasive computing," *IEEE Pervasive Comput.*, vol. 2, no. 1, pp. 46–55, Jan./Mar. 2003. doi: 10.1109/MPRV.2003.1186725.

[24] A. R. Beresford and F. Stajano, "Mix zones: User privacy in location-aware services," in *Proc. PERCOMW*, Piscataway, NJ, USA, Mar. 2004, pp. 127–131.

[25] B. Palanisamy and L. Liu, "Attack-resilient mix-zones over road networks: Architecture and algorithms," *IEEE Trans. Mobile Comput.*, vol. 14, no. 3, pp. 495–508, Mar. 2015. doi: 10.1109/TMC.2014.2321747.

[26] B. Palanisamy and L. Liu, "MobiMix: Protecting location privacy with mix-zones over road networks," in *Proc. ICDE*, Hannover, Germany, Apr. 2011, pp. 494–505.

[27] B. Palanisamy and L. Liu, "Effective mix-zone anonymization techniques for mobile travelers," *GeoInformatica*, vol. 18, no. 1, pp. 135–164, Jan. 2014. doi: 10.1007/s10707-013-0194-y.

[28] X. Liu, H. Zhao, M. Pan, H. Yue, X. Li, and Y. Fang, "Traffic-aware multiple mix zone placement for protecting location privacy," in *Proc. ICCC*, Piscataway, NJ, USA, Mar. 2012, pp. 972–980.

[29] X. Liu and X. Li, "Privacy preserving techniques for location based services in mobile networks," in *Proc. IPDPSW*, Shanghai, China, May 2012, pp. 2474–2477.

[30] Y. Sun, B. Zhang, B. Zhao, X. Su, and J. Su, "Mix-zones optimal deployment for protecting location privacy in VANET," *Peer-to-Peer Netw. Appl.*, vol. 8, no. 6, pp. 1108–1121, Nov. 2015. doi: 10.1007/s12083-014-0269-z.

[31] H. Kido, Y. Yanagisawa, and T. Satoh, "An anonymous communication technique using dummies for location-based services," in *Proc. ICPS*, Santorin, Greece, Jul. 2005, pp. 88–97.

[32] A. Pingley, N. Zhang, X. Fu, H.-A. Choi, S. Subramaniam, and W. Zhao, ''Protection of query privacy for continuous location based services,'' in *Proc. INFOCOM*, Shanghai, China, Apr. 2011, pp. 1710–1718.

[33] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan, ''Private queries in location based services: Anonymizers are not necessary,'' in *Proc. ACM SIGMOD*, Vancouver, BC, Canada, 2008, pp. 121–132.

[34] A. Khoshgozaran, C. Shahabi, and H. Shirani-Mehr, ''Location privacy: Going beyond K-anonymity, cloaking and anonymizers,'' *Knowl. Inf. Syst.*, vol. 26, no. 3, pp. 435–465, Mar. 2011. doi: 10.1007/s10115-010-0286-z.

[35] K. Mouratidis and M. L. Yiu, ''Shortest path computation with no information leakage,'' *Proc. VLDB*, vol. 5, no. 8, pp. 692–703, Apr. 2012. doi: 10.14778/2212351.2212352.

[36] X. Yi, M. G. Kaosar, R. Paulet, and E. Bertino, ''Single-database private information retrieval from fully homomorphic encryption,'' *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 5, pp. 1125–1134, May 2013. doi: 10.1109/TKDE.2012.90.

[37] A. Khoshgozaran and C. Shahabi, ''Blind evaluation of nearest neighbor queries using space transformation to preserve location privacy,'' in *Advances in Spatial and Temporal Databases*. Berlin, Germany: Springer, 2007, pp. 239–257.

[38] F. Tian, X. L. Gui, X. J. Zhang, J. W. Yang, P. Yang, and S. Yu, ''Privacy-preserving approach for outsourced spatial data based on POI distribution,'' *Chin. J. Comput.*, vol. 37, no. 1, pp. 123–138, Mar. 2014.

[39] S. Papadopoulos, S. Bakiras, and D. Papadias, ''Nearest neighbor search with strong location privacy,'' *Proc. VLDB*, vol. 3, nos. 1–2, pp. 619–629, Sep. 2010. doi: 10.14778/1920841.1920920.

[40] F. Zhang and W. W. Ni, ''Pseudo-random number encryption based location privacy preserving nearest neighbor querying,'' (in Chinese), *J. East China Normal Univ. (Natural Sci.)*, vol. 5, pp. 128–142, Dec. 2015. doi: 10.3969/j.issn.1000-5641.2015.05.011.

[41] X. J. Zhang, X. L. Gui, and Z. D. Wu, ''Privacy preservation for location-based services: A survey,'' (in Chinese), *J. Softw.*, vol. 26, no. 9, pp. 2373–2395, Aug. 2015. doi: 10.13328/j.cnki.jos.004857.

[42] A. Khoshgozaran, H. Shirani-Mehr, and C. Shahabi, ''Blind evaluation of location based queries using space transformation to preserve location privacy,'' *GeoInformatica*, vol. 17, no. 4, pp. 599–634, Oct. 2013. doi: 10.1007/s10707-012-0172-9.

[43] H.-Il. Kim, S. Hong, and J.-W. Chang, ''Hilbert curve-based cryptographic transformation scheme for spatial query processing on outsourced private data,'' *Data Knowl. Eng.*, vol. 104, pp. 32–44, Jul. 2016. doi: 10.1016/j.datak.2015.05.002.

[44] H. C. Liang, B. Wang, N. N. Cui, K. Yang, X. C. Yang ''Privacy preserving method for point-of-interest query on road network,'' (in Chinese), *J. Softw.*, vol. 29, no. 3, pp. 703–720, Apr. 2018. doi: 10.13328/j.cnki.jos.005451.

[45] H. Guo, Z. Zhang, J. Xu, N. An, and X. Lan, ''Accountable proxy re-encryption for secure data sharing,'' *IEEE Trans. Dependable Secure Comput.*, to be published. doi: 10.1109/TDSC.2018.2877601.

[46] X. Zhu, E. Ayday, and R. Vitenberg, ''A privacy-preserving framework for outsourcing location-based services to the cloud,'' *IEEE Trans. Dependable Secure Comput.*, to be published. doi: 10.1109/TDSC.2019.2892150.

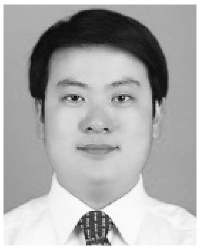
ZHAOMAN LIU was born in 1995. She is currently pursuing the master's degree in information science and engineering with Shandong Normal University. Her research interests include privacy preservation and cloud computing security.

LEI WU was born in 1980. He received the Ph.D. degree in applied mathematics from Shandong University, China, in 2009. He is currently an Associate Professor with Shandong Normal University, China. His research interests include cryptology and cloud computing security.

JUNMING KE was born in 1994. He received the master's degree from the School of Computer Science and Technology from Shandong University, Jinan, China, in 2019. He is currently a Visiting Fellow with the Singapore University of Technology and Design, Singapore. His research interests include information security and cryptography, especially blockchain and cryptocurrencies.

WENLEI QU was born in 1994. She is currently pursuing the degree in information science and engineering with Shandong Normal University. Her research interests include block chain and data security.

WEI WANG was born in 1996. He is currently pursuing the degree in information science and engineering with Shandong Normal University. His research interests include privacy preservation and fog computing.



HAO WANG received the Ph.D. degree in computer science from Shandong University, China, in 2012. He is currently an Associate Professor with Shandong Normal University. His primary interest includes public key cryptography, in particular, designing cryptographic primitives and provable security. He is currently focusing attribute-based cryptography, security in cloud computing, and blockchain.